Object Detection and Distance Ranging

Software Systems and Applications
Computer Vision

dzgf42

2018/2019

Ranging

All versions of this work iterate through a list of images and perform ranging in the same manner, namely:

1. The left and (rectified) right images are converted to greyscale.
2. They are raised to the power of 0.75 to improve following calculations.
3. OpenCV's StereoProcessor computes a corresponding disparity image.
4. Noise is reduced via OpenCV's filterSpeckles function.
5. The disparity is subsequently thresholded to a maximum value to filter inaccuracies.
6. The disparity is scaled to counteract OpenCV's scaling by 16.
7. The disparity is cropped, eliminating the area that is not covered by both cameras.
8. For each detection:
   - 8.1. We assume that the detected object will be in the foreground of its rectangle and will occupy at least 1/3 of the rectangle pixels.
   - 8.2. The general disparity image is cropped for that specific detection rectangle.
   - 8.3. The resulting disparity image array is sorted by ascending disparity
   - 8.4. Only the final third of the array is kept.
   - 8.5. The depth Z at each pixel of the remaining array is calculated with

$$Z = \frac{fB}{d}, \tag{1}$$

     where f is the focal length (pixels), B is the distance between cameras (metres) and d is the disparity at that pixel.
   - 8.6. These depths are averaged yielding a singular distance value to the detected object

Detection

v1.0 - HoG SVM with Sliding Window

For each image a window of ratio of $w : h = 1 : 2$ slides across and feeds the covered area to a trained binary SVM which classifies the input as a person or not. The process is repeated at various window scales and overlapping detections are merged via non-maximum suppression. The SVM is trained utilizing a Linear Kernel on a set of $\approx$ 24k images with a negative:positive ratio of 2:1.

v1.1 - Trying an RBF Kernel

The kernel type is changed to RBF to improve separation.

v1.2 - Preprocessing

Images are preprocessed with gamma correction before computing corresponding HoG descriptors by setting the gamma flag to true in cv2.HogDescriptor, normalizing luminance across images [1].

v1.3 - Changing Training Ratios

Due to the large amount of False Positives, it is hypthesized that perhaps the model does not have the right idea of what is not a person. As such the negative:positive ratio is increased to 3:1. A ratio of 4:1 is also considered, but the increase in training time is not met by noticeable performance improvements, and is therefore disregarded.

v2.1 - Implementing Selective Search

The image is segmented based on similarity criteria allowing the algorithm to propose regions as candidate objects after non-maximum suppression [3]. Because the proposed regions are of different ratios, and HoG requires a fixed size, sliding window search is performed on each region, so to avoid drastic distortion.

v2.3 - Final Selective Search Implementation

Because of unsatisfactory results, here a heuristic is set so that only candidate regions of $height : width \geq 2 : 1$ are accepted. This is because pedestrians will be upright and hence regions containing them will roughly be of that ratio. Furthermore, this results in minimal distortion when resizing (since the HoG fixed size is 128×64) allowing the regions' HoG descriptors to be computed directly rather than by sliding a window, improving speed.

v3.0 - Flying Pedestrian Heuristic

Another heuristic is set wherein selective search is not performed on the top portion of the image as it is very unlikely to find pedestrians there. Selective Search is also not applied to the area containing the car bonnet, although this is cropped from the image entirely.

v3.1 - Area-Depth Heuristic

Here we reason that if an object is close to the camera, then the area of their detection rectangle should be large and viceversa. Utilizing the disparity information, average human dimensions and the camera focal lengths, we can determine the expected detection area for a given depth, and filter out detections that do not align with these predictions within some "mush" factor.

v4.1 - MaskRCNN Implementation

Since SVMs are not ideal for multiple classifications, a Deep Learning route is taken via MaskRCNN [2] which utilizes the ResNet101 Neural Net trained on COCO data coupled with Selective Search to provide object detection and segmentation via semantic masking. For this particular work, the masking capabilities are disregarded. Through this integration, 80 different object classes are detectable while still performing the ranging task.

v4.2 - MaskRCNN Simple Thresholding Heuristic

To further decrease false positives, detections with confidence scores ≤ 0.9 are rejected.

Results

Fig. 1 presents example results from v3.1 and v4.2. Fig. 2 displays an evaluation of the work, presenting the setup time T_s, time-per-frame T_f, and the average percentage of false positives per image $FPPI_{\%}$ for each version. Timing was calculated through the averaging of repeated runs and OpenCV's getTickCount(). For T_f and $FPPI_{\%}$, the image list was shuffled before iterating to ensure a random distribution of images. $FPPI_{\%}$ was calculated across 23 random images from the list for each version. The total number of detections and false positives was counted across the 23 images, before dividing the latter by the former.

Figure 1: a) A sample detection and ranging performed by v4.2. We notice the high accuracy, low number of false positives and the multi-class ability. b) A sample and detection and ranging performed by v3.1. Noticeably worse, there still is a relatively low number of false positives.

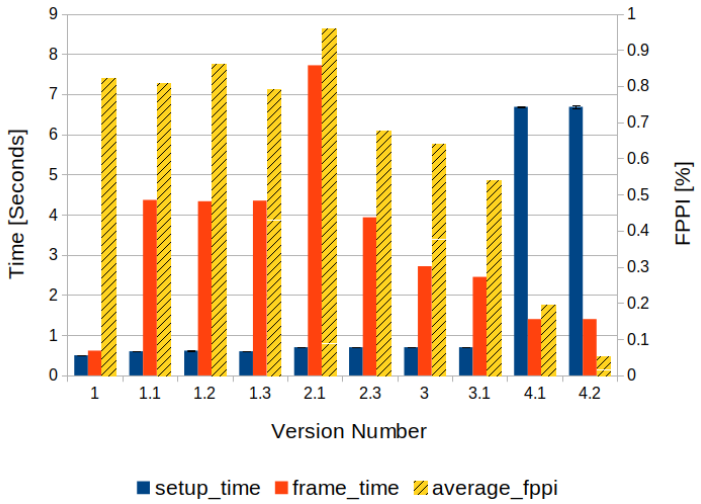

Figure 2: Bar graph illustrating the performance across the various versions. Note that Setup and Frame Time utilize the left y-axis while Average FPPI percentage utilize the right y-axis. For an ideal performance, all three quantities should be minimized. Interesting to note is the complete failure of v2.1 and the general improvement from v1.0 to v1.3 in terms of frame time and FPPI. Fig. 3 provides further detection insights.

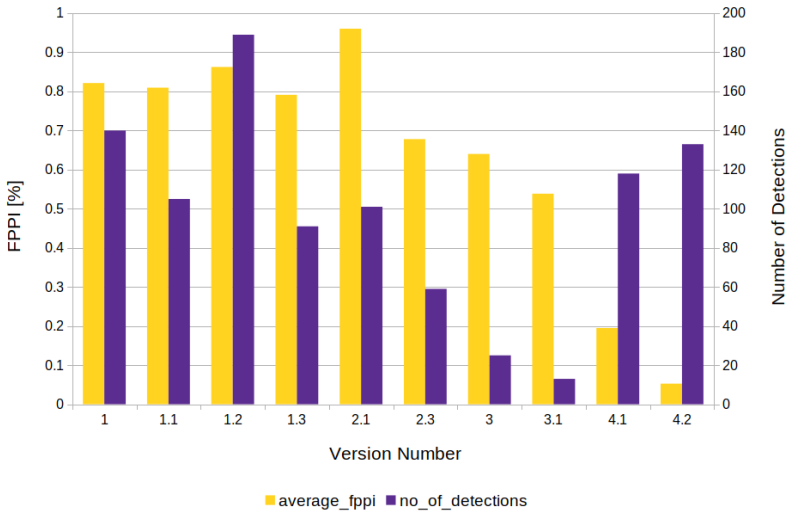

Figure 3: Bar Graph illustrating the the reltionship between Average FPPI percentage and total Number of Detections per Image.From Fig. 2, one may argue that the work improved in accuracy throughout its versions through the lower number of false positives. However, the lower number of general detections shown here should also be considered, as this is most probably hiding a large amount of False Negatives. In an ideal situation in fact, we would want The Number of Detections to be maximized while minimizing FPPI, as occurs in versions v4.x. This is of particular note in autonomous vehicles system where a more conservative system is desired.

Words in text (via TeXcount): 744.

References

[1] Navneet Dalal and Bill Triggs. "Histograms of oriented gradients for human detection". In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE. 2005, pp. 886–893.

[2] Kaiming He et al. "Mask r-cnn". In: *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE. 2017, pp. 2980–2988.

[3] Jasper RR Uijlings et al. "Selective search for object recognition". In: *International journal of computer vision* 104.2 (2013), pp. 154–171.